

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Patent Application

5 Applicant(s): Gn et al
Case: 3-3-1-1-1
Serial No.: 10/081,874
Filing Date: February 21, 2002
Group: 2179
10 Examiner: Mylinh T. Tran

Title: Method and Apparatus for Generating a Graphical Interface to Enable Local or
Remote Access to an Application Having a Command Line Interface

15

REPLY BRIEF

20 Mail Stop Appeal Brief – Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

25 Sir:

Appellants hereby reply to the Examiner's Answer, mailed July 5, 2007 (referred to hereinafter as "the Examiner's Answer"), in an Appeal of the final rejection of claims 1-22 in the above-identified patent application.

30

REAL PARTY IN INTEREST

A statement identifying the real party in interest is contained in Appellants' Appeal Brief.

35

RELATED APPEALS AND INTERFERENCES

A statement identifying related appeals is contained in Appellants' Appeal Brief.

STATUS OF CLAIMS

A statement identifying the status of the claims is contained in Appellants' Appeal Brief.

STATUS OF AMENDMENTS

A statement identifying the status of the amendments is contained in Appellants' Appeal Brief.

SUMMARY OF CLAIMED SUBJECT MATTER

A Summary of the Invention is contained in Appellants' Appeal Brief.

STATEMENT OF GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

A statement identifying the grounds of rejection to be reviewed on appeal is contained in Appellants' Appeal Brief.

CLAIMS APPEALED

A copy of the appealed claims is contained in an Appendix of Appellants' Appeal Brief.

ARGUMENT

Independent Claims 1, 8, 12 and 22

Independent claims 1, 8, 12, and 22 were rejected under 35 U.S.C. §102(b) as being anticipated by Audleman et al. In particular, the Examiner asserts that Audleman discloses querying a user to specify properties of one or more option groups provided by each of the software applications (col. 3, lines 21-20).

Appellants note that the present disclosure teaches that "the developer is then queried during step 540 to specify the properties of each *option group, i.e., for the constraints associated with a given option group, such as whether the various options within an option group can be used together and any input file requirements.*" (Page 8, lines 15-18, of the originally filed disclosure; emphasis added.) The disclosure also teaches that

the software applications have the following general syntax:

Tool_name [*option 1*] [*option 2*] <filename>

where each of these *options* further can be of one of the following types {exactly one parameter; one or more than one; none or more and with or without an input file}. In this manner, the developer 210 or administrator can establish *groups and subgroups of parameters with similar properties*.

(Page 8, line 28, to page 9, line 4, of the originally filed disclosure; emphasis added.)

Finally, the present disclosure teaches that "the developer 210 or administrator is queried using a second interface 900, shown in FIG. 9, to specify the properties of each type of option, i.e., for the *constraints associated with a given option group*, such as whether the various options (identified in window 950) within an option group can be used together in field 910 and any input file requirements in field 970." (Page 11, lines 21-24, of the originally filed disclosure; emphasis added.)

Once the tool is registered, "the web page 1100 allows the *user to specify the arguments for the input files for the various option groups*, as appropriate." (Page 12, lines 4-6, of the originally filed disclosure; emphasis added.)

In the text cited by the Examiner, Audleman teaches that

the client 104 executes a user interface program 114 that interacts with the CS 110 to provide an operator with control over the CP's 108. The user interface program 114 receives a copy of the XML files 112 from the CS 110, wherein the XML files 112 represent a command syntax of the highest active release level of the command set for the CP's 108. *The user interface program 114 processes the XML files 112 to identify resource types, verbs, and keywords*. The user interface program 114 then dynamically displays a "Wizard" comprising a step-by-step series of dialogs that guide the operator through the command syntax, and constructs one or more commands based on the operator selections from the dialogs.

(Col. 3, lines 18-30; emphasis added.)

Regarding resource types, verbs, and keywords, Audleman teaches that the general format of the command syntax for the CP 108 comprises the following:

Verb Resource type Keyword (Parameter)

Each of these elements is described below:

VERB--The verb is the first element and identifies the action to be taken. The verb can be abbreviated.

RESOURCE TYPE--The resource type is the second element and identifies the type of resource acted upon by the command. The resource may comprise, for example, transactions, databases, terminals, areas, users, etc. The

resource type may support a synonym.

KEYWORD--Keywords are optional elements, depending upon the specific command. The keyword may identify resources of one or more CP's 108. The keyword may also identify a function of the command.

5 PARAMETER--A parameter identifies a defined or created value or resource, such as a database, terminal, area, user, etc. Parameters in commands must be replaced with values. Multiple parameters are separated by a comma within the parentheses.
(Col. 4, lines 3-22.)

10 Audleman, however, does not disclose or suggest *options groups* as defined in the present invention, and does not disclose or suggest *specifying properties of options groups*. Independent claims 1, 8, 12, and 22 require *querying a user to specify properties of one or more option groups* provided by each of said software applications.

15 Thus, Audleman et al. do not disclose or suggest querying a user to specify properties of one or more option groups provided by each of said software applications, as required by independent claims 1, 8, 12, and 22.

Claims 11 and 21

20 Claims 11 and 21 are rejected under 35 U.S.C. §102(b) as being anticipated by Audleman et al. In particular, the Examiner asserts that Audleman demonstrates that the remote server script provides any necessary input files to said remote server, initiates the execution of said selected software application on said remote server and returns any results to said client (col. 2, line 55, to col. 3, line 30).

25 Appellants, however, could find no disclosure or suggestion by Audleman of a *remote server script* that provides any necessary input files to a remote server, and of initiating the execution of a selected software application on the remote server and returning any results to a client. Claims 11 and 21 require wherein said remote server script provides any necessary input files to said remote server, initiates the execution of said selected software application on said remote server and returns any results to said client.

30 Thus, Audleman et al. do not disclose or suggest wherein said remote server script provides any necessary input files to said remote server, initiates the execution of said selected software application on said remote server and returns any results to said client, as required by claims 11 and 21.

Response to Arguments

In the Response to Arguments section of the Examiner's Answer, the Examiner asserts that "the specification sets forth examples of what the options groups are, but does not limit the terminology to only those examples," and asserts that "Option groups' alone can still
5 be read in light of Audleman's teaching because VERB, RESOURCETYPE, KEYWORD, PARAMETER are, in fact, option groups provided by each of the software applications."

First, as noted above, Audleman does not disclose or suggest "option groups," where "option groups" are "*groups and subgroups of parameters with similar properties*." (See, page 8, line 28, to page 9, line 4, of the originally filed disclosure; emphasis added)

10 Second, as noted above, the present disclosure teaches that "the developer is then queried during step 540 to specify the properties of each *option group, i.e., for the constraints associated with a given option group, such as whether the various options within an option group can be used together and any input file requirements*." (Page 8, lines 15-18, of the originally filed disclosure; emphasis added) VERBS, RESOURCETYPES, and KEYWORDS
15 are not constraints and are not option groups, as defined in the context of the present invention. (See, the definitions of VERBS, RESOURCETYPES, and KEYWORDS defined by Audleman and recited above.)

In addition, in the text cited by the Examiner, Audleman teaches that

20 the client 104 executes a user interface program 114 that interacts with the CS 110 to provide an operator with control over the CP's 108. The user interface program 114 receives a copy of the XML files 112 from the CS 110, wherein the XML files 112 represent a command syntax of the highest active release level of the command set for the CP's 108. *The user interface program 114 processes the XML files 112 to identify resource types, verbs, and keywords*.
25 The user interface program 114 then dynamically displays a "Wizard" comprising a step-by-step series of dialogs that guide the operator through the command syntax, and constructs one or more commands based on the operator selections from the dialogs.
(Col. 3, lines 18-30; emphasis added.)

30 Thus, even if a PARAMETER is considered a constraint, Audleman has explicitly omitted PARAMETERS from the entities identified in the *XML files 112 processed by the user interface program 114*. In addition, a PARAMETER itself is *not* an "option group." Thus, Audleman does not disclose or suggest *specifying properties of options groups*, as required by independent

claims 1, 8, 12, and 22.

In the Response to Arguments section of the Examiner's Answer, the Examiner asserts that "Command Servers 110" are considered as the "remote server script."

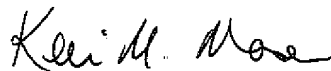
Appellants, however, could find no disclosure or suggestion in Audleman that the Command Servers 110 *initiate the execution of a selected software application* on the remote server, could find no disclosure or suggestion that the Command Servers 110 *return any results to a client*, and maintain that a Command Server is *not* a remote server *script* that provides any necessary input files to a remote server. Thus, Audleman et al. do not disclose or suggest wherein said remote server script provides any necessary input files to said remote server, initiates the execution of said selected software application on said remote server and returns any results to said client, as required by claims 11 and 21

Conclusion

The rejections of the cited claims under section 102 in view of Audleman et al. are therefore believed to be improper and should be withdrawn. The remaining rejected dependent claims are believed allowable for at least the reasons identified above with respect to the independent claims.

The attention of the Examiner and the Appeal Board to this matter is appreciated

Respectfully,



Date: September 5, 2007

Kevin M. Mason
Attorney for Applicant(s)
Reg. No. 36,597
Ryan, Mason & Lewis, LLP
1300 Post Road, Suite 205
Fairfield, CT 06824
(203) 255-6560

APPENDIX

1. A method for generating a graphical interface for one or more software applications having a command line interface, said method comprising the steps of:

5 querying a user to specify properties of one or more option groups provided by each of said software applications; and

 generating a graphical user interface based on said specified properties for each of said software applications, said graphical user interface identifying each of said software applications and allowing a selected one of said software applications to be accessed.

10 2. The method of claim 1, wherein said properties of each option group includes an indication of whether the various options within an option group can be used together.

15 3. The method of claim 1, wherein said properties of each option group includes an indication of any input file requirements.

 4. The method of claim 1, wherein said properties of each option group includes a name of a corresponding software application.

20 5. The method of claim 1, wherein said properties of each option group includes a location of a corresponding software application

25 6. The method of claim 1, wherein said graphical user interface allows a client to access a selected software application without regard to a location of said selected software application.

 7. The method of claim 1, wherein said graphical user interface presents a client with only valid options for a selected software application.

8 A method for enabling remote access to one or more software applications having a command line interface, said method comprising the steps of:

querying a user to specify properties of one or more option groups provided by each of said software applications; and

5 generating a graphical user interface based on said specified properties for each of said software applications, said graphical user interface identifying each of said software applications and allowing one or more clients to remotely access a selected software application

9 The method of claim 8, wherein a central server interacts with said one or
10 more clients and a remote server where said selected software application is located.

10 The method of claim 9, wherein said central server interacts with said one or more clients and said remote server using a remote server script.

11 The method of claim 10, wherein said remote server script provides any
15 necessary input files to said remote server, initiates the execution of said selected software application on said remote server and returns any results to said client.

12 A system for generating a graphical interface for one or more software
20 applications having a command line interface, said system comprising:

a memory that stores computer-readable code; and

a processor operatively coupled to said memory, said processor configured to implement said computer-readable code, said computer-readable code configured to:

25 query a user to specify properties of one or more option groups provided by each of said software applications; and

generate a graphical user interface based on said specified properties for each of said software applications, said graphical user interface identifying each of said software applications and allowing a selected one of said software applications to be accessed.

13. The system of claim 12, wherein said properties of each option group includes an indication of whether the various options within an option group can be used together.

5 14. The system of claim 12, wherein said properties of each option group includes an indication of any input file requirements

15. The system of claim 12, wherein said properties of each option group includes a name of a corresponding software application.

10 16. The system of claim 12, wherein said properties of each option group includes a location of a corresponding software application.

15 17. The system of claim 12, wherein said graphical user interface allows a client to access a selected software application without regard to a location of said selected software application.

18. The system of claim 12, wherein said graphical user interface presents a client with only valid options for a selected software application.

20 19. The system of claim 12, wherein a central server interacts with one or more clients and a remote server where said selected software application is located.

25 20. The system of claim 19, wherein said central server interacts with said one or more clients and said remote server using a remote server script.

30 21. The system of claim 20, wherein said remote server script provides any necessary input files to said remote server, initiates the execution of said selected software application on said remote server and returns any results to said client.

22. An article of manufacture for generating a graphical interface for one or more software applications having a command line interface, comprising:

a computer readable medium having computer readable code means embodied thereon, said computer readable program code means comprising:

5 a step to query a user to specify properties of one or more option groups provided by each of said software applications; and

a step to generate a graphical user interface based on said specified properties for each of said software applications, said graphical user interface identifying each of said software applications and allowing a selected one of said software applications to be accessed.

10

EVIDENCE APPENDIX

There is no evidence submitted pursuant to § 1 130, 1 131, or 1.132 or entered by the Examiner and relied upon by appellant.

RELATED PROCEEDINGS APPENDIX

There are no known decisions rendered by a court or the Board in any proceeding identified pursuant to paragraph (c)(1)(ii) of 37 CFR 41.37